R Basics

Esteban Lopez Ochoa, PhD
University of Texas at San Antonio

Jan 20, 2022

Overview

This exercise provides an overview of the main characteristics of `R` language and syntax. We will use the IDE (Integrated Development Environment) version of `R`, `R-Studio` as our main platform. We will become familiar with the main parts of `R-Studio`; learn the basic language; create projects, objects and even generate a figure.

A number of tasks must be performed in order to complete this exercise:

1. Open `R-Studio` (or the non IDE version - `R`), become familiar with the different parts and menus of the software.
2. Run basic commands to get familiar with the `R` Language. Specifically, learn to do the following:
    - Clear memory, Set a seed, and Save Input and output
    - Manage a session work space, create an R-Studio project, and Set work directories
    - Use as a basic calculator
    - Create different types of objects

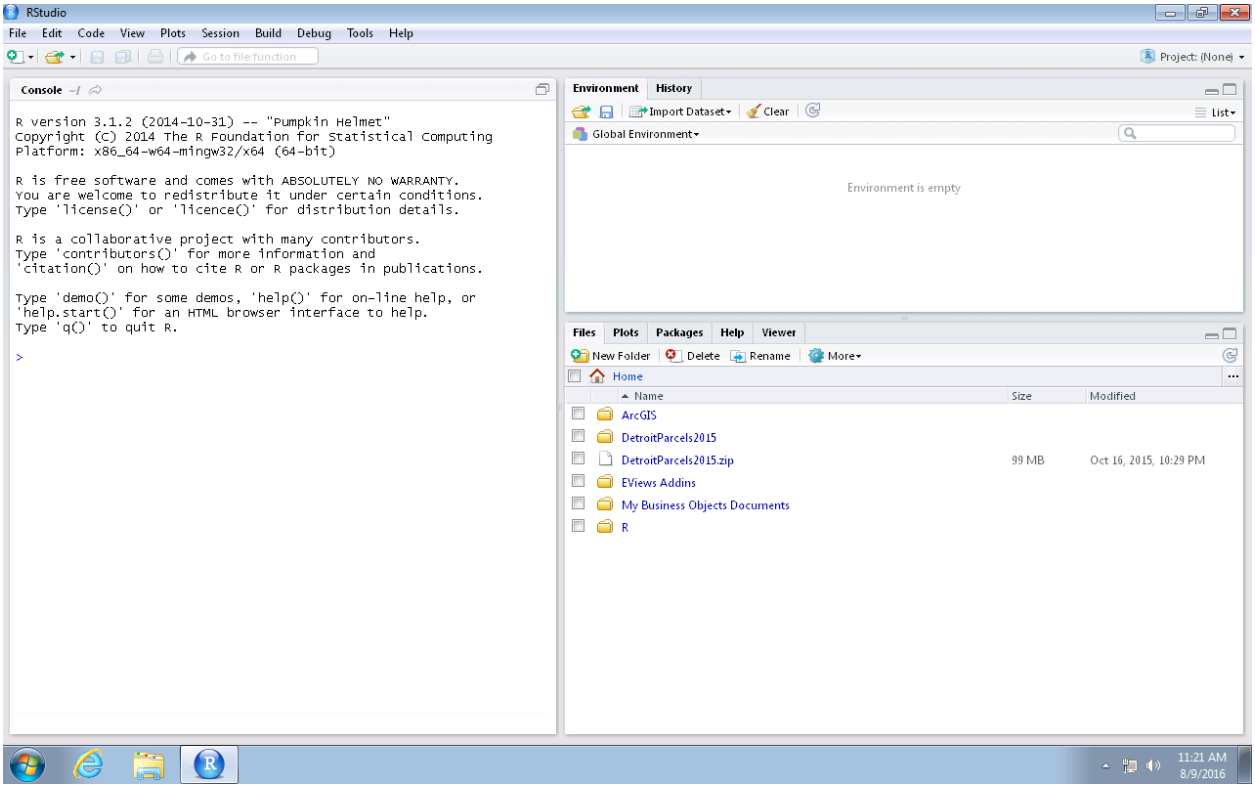
    Lab 1 assignment:

Once you are done reading this introductory guide, please read the document *"Assignment_Instructions.pdf"* to learn how to complete your first Lab assignment. Due on Sunday Jan 23rd, 2020 by 23:59.

Part 1. Open R-Studio, become familiar with the different parts and menus of the software.

Step 1: Open R-Studio. As the Image 1 shows bellow, there are three panels: Console, one starting with Environments, and another one starting with Plots.

Image 1.



The Console is the main processing window or the *brain* of R-Studio (which is R). Here you can directly type any command followed by pressing the *Return* key in your keyboard, and then R will process your request. Run the following code (by copy-pasting each line at a time and then hitting the return key) in your R-Studio console pane, and see what is the output:

```
2+2
"I am nervous about learning R, but that is OK. Hello R!!"
```

As you can see, R only prints the output. In the first case it actually processed the sum you instruct it. In the second case, it only printed the phrase since it is just that, text.

The Environment tab shows a list of what R has stored in the current session's memory. If your create an object, or load a data set, these will appear here. Run the following code in your R-Studio session, and see what is the output. Check if the Environment tab has changed.

```
result <- 2+2
result
```

A fourth panel in R-Studio is the Source. This is the place where you type R commands and store them for future replication in a file that is usually called an R script. Different from other software with drop-down menus that you will easily forget later, one of the main advantages of R is the ability to trace back what you have done and be able to replicate it. By writing a list of commands in R that achieve a certain output

(say a pretty Map or a Table of results) you can always go back and replicate that figure following that set of steps written in your `R script`. Besides writing `R` commands, you can also write comments about the commands you write by starting a line with the pound sing `#`. For example

```
# What is 2 + 2?
2+2
```

To create a new script and hence activate the `Source` panel click in **File > New File > R Script**.

Now, the usual dynamics when working in R studio is that you will write some commands in an `R script` and then run them into the `Console` by typing *Ctr + Return* in Windows or *Cmd + Return* in Mac. Hence let's re-order the panels so we can have a more proper working environment. Click in **Tools > Global Options > Pane Layout** and change the order of these panels to resemble Image 2 bellow. Then click *OK*. Finally, minimize the `Environment` and `Plots` panel to maximize the space for the `Console` and `Source` panels respectively.

Image 2.

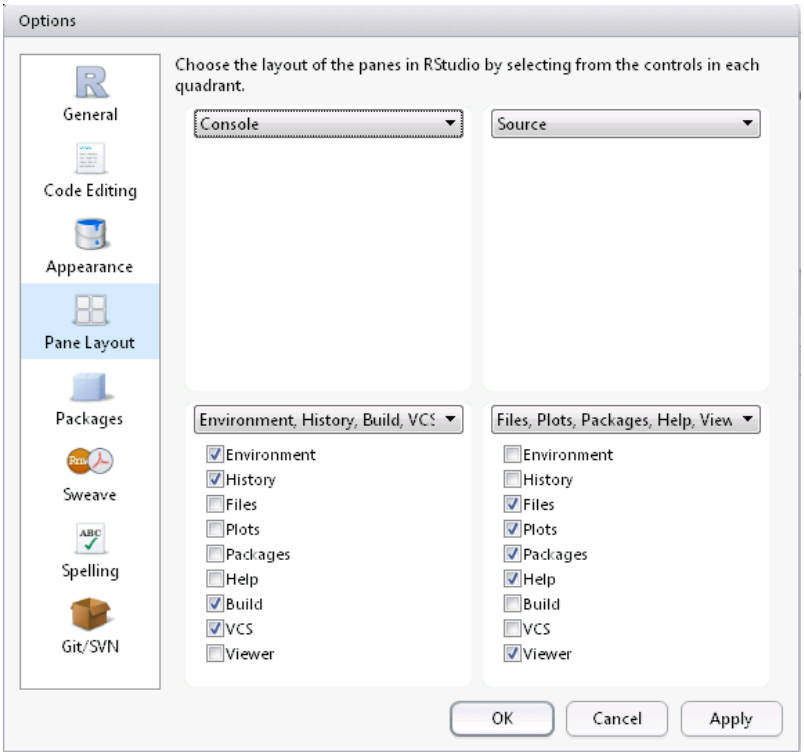
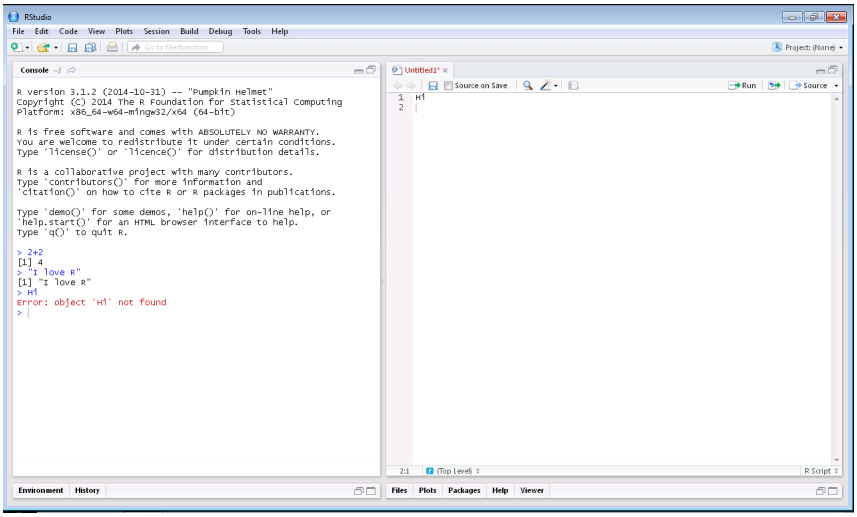
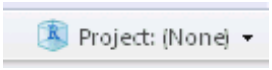


Image 3.



Part 2. Run basic commands to get familiar with the R Language.

Manage a session work space, create an R-Studio project, and Set work directories A directory is the folder in your computer hard-drive where R will save and load datasets. There are several ways to set directories. The easiest way is by creating an R-Studio project. This will open a dialog on which you can select the folder you would like to work on. This can be done by clicking on the *Project (None)* icon and then *New Project*.



It is strongly recommended that you chose a place within your computer to keep it throughout the semester so you can have all your files in the same place. Create a folder in *My Documents* with the name *abc123-URP5393Labs*. Then every time you start R-Studio you can create a new project with the name *01Lab*. After you are done, as you can see, the tab Files now has located the project Folder and shows the contents of it. As you can see, the previous script that you had opened has disappeared. Although you could recover that script by simply closing the current project, let's just open a new one by typing *Crtl + Shift + N*.

The following two functions are another useful way to set directories and check where things are being saved or loaded:

By typing `get_wd()` (get working directory) in the console, you will get the *path* of the current folder where R-Studio will look for to load and save files. By doing this step, you can confirm that the output of `get_wd()` is the same folder you defined in your project.

Finally, `set_wd()` (yes! you guessed, it means: set working directory) is the command that allows you to set a new working directory overriding any working directory that R-Studio might be on. This could be useful when working outside of a defined project or when creating suborders in your project folder for saving Figures or Tables. But we won't use this for now.

Use as a basic calculator Copy the following lines in a new Script and run them into the Console one by one. You can achieve this in two different ways:

1. Place the text cursor (blinking vertical line) in the line of the command you want to run and type *Crtl + Return*.
2. You can Click the *Run* button located at the upper-right corner of the Source panel.

```
2+2
3-1
4*6
6/2
sqrt(20)
```

Create different types of objects R is an object oriented language, which means that you can create different types of objects and work with them to achieve your results. The most basic object creation is the following:

```
a<-1
```

In this line we have assigned (using the <- symbol) the value of 1 to the letter a. Now every time you type only a in your Console, R will return the value of 1.

Now if we create another object, say b and assigned a different value, we can use these objects to create a third one:

```
b<-2
c<-a+b
c
```

```
## [1] 3
```

Just to check, if you type a different type of letters in the same fashion as the example bellow, such as g<-d + e, you will see that R doesn't understand what you are trying to do and tells you that there is no d object in its memory to work with.

The following table provides a non-exhaustive list of the most common objects we will use in this course

Object	R command	Example
numeric	none	a<-1
string	" "	b<- "You know nothing JS"
list	list()	l1 <- list(a,b)
matrix	matrix()	m1<- matrix(0,2,2)
sequence	seq()	sq1<-seq(1,10,1) and sq2<-letters[sq1]
dataframe	data.frame()	df1<-data.frame(sq1,sq2)